AF/2124
23

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of: Sai V. Allavarpu, Rajeev Angal, Toney T. Vuong	§	Group Art Unit: 2124
	§	Examiner: Masinick, Michael D.
	§	Atty. Dkt. No.: 5181-48500 P4505
Serial No. 09/553,970	§	
Filed: April 21, 2000	§	
For: Thread-Safe Remote Debugger	§	

CERTIFICATE OF MAILING
37 C.F.R. § 1.8

I hereby certify that this correspondence is being deposited with the U.S. Postal Service with sufficient postage as First Class Mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on the date indicated below:

Robert C. Kowert
Name of Registered Representative

August 12, 2004 [illegible]
Date Signature

REPLY BRIEF

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir/Madam:

In response to the Examiner's Answer mailed June 16, 2004, Appellant presents this Reply Brief. Appellant respectfully requests that this reply brief be entered pursuant to 37 C.F.R. § 1.193(b)(1) and considered by the Board of Patent Appeals and Interferences.

REPLY TO EXAMINER'S ANSWER

Related Appeals and Interferences

The Examiner states (Examiner's Answer, item 2) that Appellants' appeal brief does not contain a statement identifying the related appeals and interferences which directly effect or will be directly affected by or have a bearing on the decision in the pending appeal. Appellants, however, point to section II, page 2, of Appellants Appeal Brief, dated March 23, 2004, which specifically states, "[n]o other appeals or interferences are known which would directly affect or be directly affected by or have a bearing on the Board's decision in this appeal."

Response to Examiner's Answer

A. Claims 1, 9, 10, 13-15, 17-22, 30-32, 35-37, 39-44, 52, 53, 56-58, and 60-64

Please note that in the Appeal Brief, claims 10, 31, and 53 where erroneously listed both as part of this claim grouping and as standing or falling alone. However, Appellants intended for claims 10, 31, and 53 to be part of this claim grouping and no arguments were presenting regarding claims 10, 31, and 53 standing or falling alone.

In the Appeal Brief, Appellants have argued that Wygodny fails to teach a thread-safe debug service which is executable on a client computer system and a thread-safe remote control service that is executable on the client computer system <u>to receive control requests from an external source</u> to initiate and manage the debug service on the client computer system. Wygodny teaches an analysis system including two different modes of operation, an online mode and a remote mode. In Wygodny's online mode, a developer can "locally analyze a client program" (Wygodny, column 6, line 59). During online mode, both the client program and Wygodny's Analyzer tools are running on the same computer, hence *locally* analyzing a client. In remote mode, a developer creates a TCI file of preset tracing instructions (Wygodny, column 5, lines 28-31) and supplies the TCI file along with a bugtrapper agent to a remote customer who then executes the agent and the client application and sends a generated trace file back to a developer via email for post execution analysis (Wygodny, column 3, lines 29-39).

In response to Appellants arguments, the Examiner argues that Wygodny's Bugtrapper 104 is on the Agent side, as illustrated in Figure 1B, and thus remotely located from traced application 102 on the client side. For example, the Examiner states, "[I]tem 104 send commands (provides remote control) to the remote site (top line from 104) to control execution ... at the Client side", (parenthesis in original), referring to figure 1B of Wygodny. (Examiner's Answer, page 4, lines 6-9). Apparently the Examiner is equating the agent side in Fig. 1B of Wygodny to an external source sending control requests to the client side to initiate and manage the debug services on the client side. However, **all objects depicted in Fig. 1B are located at the customer site (i.e. on the same computer)**, as is indicated by the underlined caption "At the customer site" in Fig. 1B. Thus, Wygodny's Bugtrapper agent cannot be an external source sending control requests. Following the Examiner's line of argument, the client side, as illustrated in Figure 1B, would have to equate to the thread-safe remote control service of claim 1. This is clearly an incorrect interpretation of Wygodny. The "client side" refers to the *traced user application,* which is identified as the "client" in Fig. 1B and at col. 5, lines 30 – 31, while the "agent side" refers to the bugtrapper agent 104. Both the traced user application (client) and the bugtrapper (agent) run together on the customer's computer and thus Fig. 1B of Wygodny does not illustrate a thread-safe remote control service which is executable on the client computer system to receive control requests from an external source to initiate and manage the debug services on the client computer system.

In response to Appellants arguments regarding the various processes illustrated in Figure 1B executing on the same computer, the Examiner states, "it is not clear why this is even considered an issue" (Examiner's Answer, page 5, lines 6-7). The fact that both the agent side and client side of Figure 1B (and Figure 2) execute on the same computer is relevant to refute the Examiner's contention that Wygodny's Bugtrapper 104 is an external source sending command to the remote site to control execution on the client side (See Examiner's Answer, page 4, lines 6-8). Clearly, if both the Bugtrapper and client application are executing on the same computer, the Bugtrapper 104 cannot be an external source sending control requests as the Examiner assumes.

The Examine further contends that the customer site in Wygodny can be considered to be one or more computers. This is pure speculation by the Examiner as Wygodny very clearly illustrates and describes, in detail, how both the bugtrapper and client application execution on the same computer. Wygodny clearly teaches this (Wygodny, column 16, lines 17-25) and further states, "[t]he agent 104 and the client-side trace library 125 <u>run in the same context</u> … [f]or the purposes herein, a context can be a process, a thread, or any other unit of dispatch in a computer operating system" (Wygodny, column 5, lines 54-60). Wygodny thus **<u>expressly teaches</u>** that both the agent side and client side run in the same process or thread in a computer system and therefore that the bugtrapper agent is <u>not</u> an external source sending control requests to the client user application.

The Examiner responds to this argument by stating, in reference to Figure 2, "[i]f both the client side and agent side were considered the same computer, there would be no need to label each side" (Advisory Action dated January 29, 2004, and Examiner's Answer, page 6, lines 11-12). This argument makes no sense and is clearly based completely on the Examiner's assumptions. Appellants assert that it is very well known and common practice to illustrate multiple modules, objects, processes, etc, running on the same computer using separately labeled blocks. The Examiner further argues, again referring to Figure 2, "the sides have shared memory 105 of fig. 2… if they were on the same computer, all the memory would be shared" (Advisory Action dated January 29, 2004, and Examiner's Answer, page 6, lines 12-14). This argument is clearly incorrect. The term "shared memory" in Wygodny specifically refers to memory shared between processes executing on the same computer. Additionally, it is also very well known that the memory of a computer system is routinely segmented among executing processes, specifically to prevent the processes from accessing, and thus possibly corrupting, the memory of other processes. The shared memory in Wygodny in no way implies that the Bugtrapper agent (agent side) executes on a separate computer from the traced application (client side).

In response to Appellants arguments that Wygodny's Bugtrapper does not receive requests from an external source, the Examiner appears to change his interpretation of Wygodny midstream by stating, "the developer is considered the external source" (Examiner's Answer, page 6, lines 19-22). Appellants submit, however, that a developer issuing commands through a user interface when locally analyzing a client in Wygodny's online mode cannot be considered an external source sending control requests. Furthermore, as shown above, and in Appellants Appeal Brief, Wygodny's remote mode uses only predefined, static, tracing instructions from a TCI file and therefore also does not involve receiving control requests from an external source.

Regarding the Examiner's arguments involving dictionary definitions of the terms, "remote" and "online", Appellants submit that these particular definitions are irrelevant because Wygodny provides extremely detailed descriptions of both the functionality and mechanisms used in both his "remote mode" and "online mode." Thus, whatever other definitions of the terms may exist, Wygodny is using them only as names for the two modes of his system and does not rely upon them to define the scope or functionality of his system. Interpreting Wygodny's system solely in terms of a particular dictionary's definitions of these two terms contrary to, and to irrespective of, Wygodny's expressed teachings is clearly improper. Appellants further submit that Wygodny is using the term "online mode" to refer to a live, running debug session on the developer's computer, in contrast to his "remote mode" where the developer can only examine the previously saved trace output sent by a customer. Furthermore, the Examiner seems to be improperly combining Wygodny's remote and online modes into a single system, picking and choosing from each as needed for his arguments.

The Examiner also refers to Wygodny's summary (Wygodny, column 2, lines 53-56) that describes a software system that allows a developer to trace the execution paths of a client application. The Examiner maintains that this statement *inherently* includes allowing the developer to interactively and *remotely* control the client application (Examiner's Answer, page 4, lines 11-14). Appellants note however that Wygodny, at the Examiner's cited passage, is describing how a developer generates the TCI file that contains the tracing instructions subsequently used to capture the client program's execution path. As described above, the TCI file is either used by the developer to *locally* analyze a client program, or is sent to a user for use in Wygodny's remote mode where the agent program and the client-side trace library use the information in the TCI file to determine what information to collect from a traced client program. (Wygodny, column 6, lines 3-5). Wygodny also teaches that the user executes the agent and can cause the agent to dump the trace data to a disk file that can be sent back to the developer (Wygodny, column 5, lines 40-41, and column 6, lines 7-12). Thus, in remote mode, Wygodny teaches using a preset file of instructions to control the tracing of a client program, and thus cannot inherently include interactively and remotely controlling the client application, as the Examiner argues.

Appellants also submit that the Examiner's comments regarding Wygodny's system allowing a developer to trace multiple threads, in either real-time or near real-time, do not illustrate any "remote control" (See, Examiner's Answer, page 4, lines 16-20). In contrast, the ability to trace multiple threads refers only to the fact that Wygodny's Bugtrapper and Analyzer can trace multi-threaded applications, but does not infer any sort of remote control capabilities. In fact, Appellants submit that remotely controlling the execution of a program in real-time in Wygodny, as the Examiner contends, would not be possible given the latencies involved with network messaging. Wygodny specifically teaches that in remote mode, "the *agent 104 is provided to the user 110* as a stand-alone component that enables the user to generate a trace log file that represents the execution of the client" (Wygodny, column 6, lines 21 – 26) and further states, "*[f]rom the perspective of the remote user, the agent 104 acts essentially as a black box* that records the execution path of the client 102." (Wygodny, column 6, lines 38 – 45). Given Wygodny's reliance on the generation and sending of both the TCI file and the resultant trace log file, Wygodny is clearly not teaching interactive remote control of a client application from an external source, as the Examiner maintains.

Appellants further submit that the Examiner is applying multiple incompatible approaches to interpreting Wygodny during his rejection of Appellants' claims. For instance, in his rejection of claim 1, the Examiner first equates Wygodny's Bugtrapper Agent with Appellants' thread-safe remote control service "executable on the client computer system to receive control requests from an external source" (Examiner's Answer, page 3, lines 17-20). Thus, the Examiner is asserting that Wygodny's Bugtrapper Agent executes on the client computer system and receives control requests from an external source. However, the Examiner goes on to refute his own assertion by also arguing, in the same rejection of claim 1, that Wygodny's Bugtrapper Agent executes on a separate computer from the client application and sends commands to the client program, even erroneously relying on the labeling of Figure 2 and the use of shared memory to support his second interpretation of Wygodny's Bugtrapper Agent (see, Examiner's Answer, page 4, lines 4-9 and page 6, lines 7-17). Applicants further note, as shown above, that Wygodny unequivocally states that the Bugtrapper Agent and the client application execute within the same context (i.e. the same process or thread) and thus certainly on the same computer (Wygodny, column 16, lines 17-25, and column 5, lines 54-60).

After contradicting his own argument regarding the Bugtrapper Agent, the Examiner never offers any other assertion or interpretation of Wygodny that would provide for a remote control service executable on the client computer system and that receives control requests from an external source. The Examiner does however, in his argument regarding claim 9, return to his original interpretation and relies upon the Bugtrapper Agent as the remote control service executable on the client computer system.

Hence, the Examiner's stated rejection requires that Wygodny's Bugtrapper Agent be both the remote control service receiving commands from an external source and the external source sending those commands. This would obviously require that the Bugtrapper Agent execute both on and external to the client computer and to simultaneously both send and receive the same control commands. These two interpretations are completely incompatible with each other and clearly not supported in the teachings of Wygodny. Additionally, as stated above, the Examiner later argues, that the developer is an external source sending control commands (Examiner's Answer, page 6, lines 19-22). This is inconsistent with the Examiner's previous reliance on the Bugtrapper Agent as the external source. The on-line developer mode of operation is also a completely different mode of operation in Wygodny than the remote mode that the Examiner previously relied upon. Thus, the Examiner alternately argues various contradictory and improper interpretations and characterizations of Wygodny's system as needed to make individual arguments without ever providing a single, clear argument outlining how Wygodny anticipates every feature of Appellants' claims. Wygodny clearly fails to anticipate a thread-safe debug service which is executable on a client computer system and a thread-safe remote control service that is executable on the client computer system to receive control requests from an external source to initiate and manage the debug service on the client computer system.

B. Claims 2-4, 23-25, and 45-47

In the Appeal Brief, Appellants have argued that Wygodny fails to teach a debug print function which is operable independently of the remote control service, wherein the thread-safe debug print function is operable to provide debug output for one or more threads of the multi-threaded application such that the debug output of each of the one or more threads remains distinct from the debug output of the other threads.

The Examiner currently contends, according to the Grounds of Rejection section of Examiner's Answer (p. 8, lines 2-8), that Wygodny provides display information to be viewed online or saved and further argues that "displaying is considered to be either on a monitor or any other output display device" and concluding that a "print function is considered inherent to enable files to be analyzed later ... to enable issues to be resolved promptly." This is pure conjecture by the Examiner.

Printing an already saved file containing trace output information is clearly not a *debug print function*. Appellants assert that such a debug print function is not necessarily present, and thus not inherent, in displaying, to whatever output device, a previously saved file containing collected trace information. Further, the Examiner has failed to show how the saving of trace output under Wygodny operates independently of the remote control service. The Examiner, as shown above, equates Wygodny's BugTrapper Agent with a remote control service (See Examiner's Answer, page 3, Grounds of Rejection, Claim 1). Wygodny clearly teaches that it is the BugTrapper Agent that saves trace output to the trace log file (See items 106, 124, and 122 of Figure 2, column 3, lines 33-38, column 6, lines 7-11), the printing of which the Examiner currently equates to a debug print function.

C. Claims 5, 26 and 48

In the Appeal Brief, Appellants have argued that Wygodny fails to teach a debug print function which is operable to provide debug output for one or more threads of the multi-threaded application, wherein the debug output of each of the one or more threads may be directed to an output target and wherein the output target comprises a standard output terminal. In his Answer, the Examiner states, "Wygodny also teaches directing the debug output to an debug output target comprising a file, a standard output terminal, or a recipient computer system with a plurality of remote diagnostic tools" (Examiner's Answer, page 8, Claims 3-7). However, the Examiner has failed to consider that Wygodny is teaching that a developer "may use a trace analyzer program to decode the trace information into a human-readable form, ... and displays translated trace information on the display screen" (Wygodny, column 3, lines 13-18). Wygodny clearly states, "the trace itself is not displayed on the screen, but ... the user can dump the trace data to the trace log file" (Wygodny, column 6, lines 41-44). Wygodny further teaches that the trace log file is "written using an encoded format that is not readily decipherable by the user" (Wygodny, column 6, lines, 28-29) and that an analyzer application converts the encoded trace data back into "a source level format" and provide the developer with various debugging options (Wygodny, column 7, lines 39-53).

Thus, the trace data in Wygodny is not directed to a standard output terminal, but instead, saved to a file in an encoded form, translated, and only then displayed for a developer. Such a system clearly does not include a debug print function providing debug output directed to an output target comprising a standard output terminal.

D. Claims 6, 27, and 49

As Appellants have argued in the Appeal Brief, Wygodny fails to teach a debug print function which is operable to provide debug output for one or more threads of the multi-threaded application, wherein the debug output of each of the one or more threads may be directed to an output target and wherein the output target comprises a recipient computer system coupled to the client computer system. The Examiner cites Wygodny, column 3, lines 33-44 in support of his contention that Wygodny teaches directing debug output to an output target comprising a recipient computer system. Appellants note, however, that the cited passage describes how a trace file, containing encoded trace output, is "sent to the developer site (such as by email)" (parenthesis in original – Wygodny, column 3, lines 38-40). Under Wygodny, trace data collected by the client-side library is written to a trace buffer that, on command from a user, the agent copies to a trace log file. The user sends the trace log file back to the developer. (See, Wygodny, column 6, lines 5-12). Hence, the output of Wygodny's tracing is directed to a trace buffer on the same computer as the client application, not a recipient computer system coupled to the client computer system. Further, the recipient's computer in the Examiner's argument is clearly not coupled to the client computer system. Saving a trace log file and then emailing it to a developer is clearly not a debug print function providing debug output directed to an output target comprising a recipient computer system coupled to the client computer system.

E. Claims 7, 28, and 50

In the Appeal Brief, Appellants have argued that Wygodny fails to teach a debug print function which is operable to provide debug output for one or more threads of the multi-threaded application, wherein the debug output of each of the one or more threads may be directed to an output target and wherein the output target comprises a plurality of remote diagnostic tools. In his Grounds of Rejection section of his Answer, p. 8, lines 10-12, the Examiner states that Wygodny teaches directing debug output to an output target comprising a recipient computer system with a plurality of remote diagnostic tools. In addition to the arguments regarding claims 6, 27, and 49, in section D above, Appellants also assert that Wygodny allowing a developer to read, decode, and analyze the trace information saved by a client in a trace log file and emailed to deliver does not constitute a debug print function providing debug output directed to an output target comprising a plurality of remote diagnostic tools.

F. Claims 8, 29, and 51

According to the Final Office Action, mailed October 22, 2003, the Examiner rejected claims 8, 29, and 51, "as being unpatentable over Wygodny in view of the applicant['s] choice of languages used to implement his invention" (Final Office Action, page, 4, lines 4-6). In response, Appellants have argued in the Appeal Brief that this rejection is improper as the Examiner is attempting to reject the claims in view of Applicants' own work. On its face, the rejection is made over Wygodny in view of Applicants' own teachings. It is well settled law that a claim cannot be rejected based on the applicants' own teachings.

In the Grounds of Rejection section of his Answer, page 9, lines 2-7, the Examiner has improperly changed this rejection to a new rejection based on Wygodny in view of what is well known in the art. The Examiner admits that Wygodny does not teach a thread-safe debug print function operable to provide debug output in a plurality of regional or national languages, but also states, "[h]owever, the applicant does not teach the feature either ... [h]e merely relies on what was known in the art at the time of the invention" (Examiner's Answer, page 9, lines 8-9). Appellants assert however that claims 8, 29, and 51, as well as the Specification clearly describe a thread-safe debug print function is operable to provide debug output in a plurality of regional or national languages (See e.g. Specification, page 23, lines 18-24). This is Applicants' own teachings, not what is well known in the art. Appellants assert that a thread-safe debug print function operable to provide debug output in a plurality of regional or national languages is <u>not known in the prior art</u>. The Examiner has not provided any references teaching that it was known in the prior art to provide thread-safe debug print function operable to provide debug output in a plurality of regional or national languages. This rejection is completely unsupported by any evidence of record.

Appellants also assert that not only does Wygodny fail to teach or suggest presenting his trace information in anything other than C++ as the Examiner originally stated (See Office Action dated May 5, 2003, page 8, Claim 8), Wygodny also fails to teach or suggest any internationalization, or the use of regional or national language, in his system. Nor has the Examiner provided any other <u>evidence</u> in this regard. Thus, the Examiner is clearly applying hindsight analysis.

G. <u>Claims 11, 33, and 54</u>

In the Appeal Brief, Appellants have argued that Wygodny fails to teach wherein the remote control service is operable to allow a remote source to switch the debug services on and off for each of the corresponding components of the multi-threaded application by referencing each of the corresponding debug objects by name. As described above, Wygodny fails to teach a remote control service. Further, Wygodny also fails to teach allowing a *remote source* to switch the debug services on and off. In contrast, Wygodny teaches that a TCI file contains "instructions to the client-side trace library regarding the trace data to be collected" (Wygodny, column 6, lines 3-5). Hence, tracing is turned on or off locally according to the instructions contained in the TCI file as pre-determined by the developer.

In the Grounds of Rejection section of his Answer, page 8, lines 15-18, the Examiner claims that Wygodny provides checkboxes allowing a developer to switch debug services on and off for a set of components by referencing the debug objects by name. However, the Examiner's cited passage (Wygodny, column 14, lines 40-42, and item 502 of Figure 5) is describing how to use Wygodny's analyzer to create a TCI file. As shown above, the TCI file is sent to a user and provides static tracing instructions to the agent. Furthermore, both the analyzer and the client program are running on the same computer (See e.g. Wygodny, column 16, lines 17-25, column 5, miles 54-60, and column 6, lines 3-5). Thus, Wygodny clearly fails to teach that the remote control service is operable to allow a remote source to switch the debug services on and off for each of the corresponding components of the multi-threaded application by referencing each of the corresponding debug objects by name. Therefore, the rejection of claims 11, 33, and 54 is not supported by the teachings of the cited reference and reversal of the rejection of these claims is respectfully requested.

H. Claims 12, 34, and 55

In the Appeal Brief, Appellants have argued that Wygodny fails to teach wherein the remote control service is operable to allow a remote source to switch the debug services on and off for a set of the corresponding components of the multi-threaded application by specifying a pattern to select a set of the debug objects by name.

In response the Appellants arguments, the Examiner, in the Grounds of Rejection section of his Answer, page 8, lines 15-21, cites column 2, line 59 – column 3, line 4 of Wygodny and specifically refers to Wygodny "discussing the specification of a custom TCI file to pattern the debug service performed" (Examiner's Answer, page 8, lines 20-21). The cited passage discusses how the TCI file, which contains the trace instructions used by the agent and tracing library, is generated by a developer using a trace options editor that displays source code and presents controls that allow the developer to specify the code and data elements. Appellants note, however, that the cited passage, as well as the entirety of Wygodny, fails to mention the use of a pattern to select a set of debug objects by name, as the Examiner contends. Further, Appellants can find no teaching in Wygodny regarding the use of character wildcards or other pattern techniques for identifying a set of the debug objects by name.

CONCLUSION

For the foregoing reasons submitted in the Appeal Brief and this Reply Brief, it is submitted that the Examiner's rejections of claims 1 – 15, 17 – 37, 39 – 58 and 60 – 64 were erroneous. Reversal of the Examiner's decision is respectfully requested.

Respectfully submitted,

[illegible]

Robert C. Kowert
Reg. No. 39,255
Attorney for Appellant

Meyertons, Hood, Kivlin, Kowert & Goetzel, P.C.
P.O. Box 398
Austin, TX 78767-0398
(512) 853-8850

Date: August 12, 2004